Hybrid Model for Intrusion Detection using Data Mining Techniques

GRADUATE PROJECT REPORT

Submitted to the Faculty of
The School of Engineering & Computing Sciences
Texas A&M University-Corpus Christi
Corpus Christi, TX

in Partial Fulfillment of the Requirements for the Degree of
Master of Science in Computer Science

by

Himavarsha Surabi
Spring 2014

Committee Members

Dr. Longzhuang Li
Committee Chairperson

Dr. Mario Garcia
Committee Member

Dr. David Thomas
Committee Member
ABSTRACT

Due to increase in computer usage and computer based information systems, the security of networks is considered as a primary issue. The computers are being exposed to several new attacks and malicious intrusions over the internet. Intrusion detection systems (IDS) are in demand as these are one of the effective ways for detecting malicious attacks and intrusions. Intrusion detection systems are being widely used to preserve the confidentiality, integrity and availability of the information. Various Data mining techniques are being used for intrusion prevention and detection by analyzing large network traffic. Data mining techniques are used for processing large volumes of data and to detect unknown patterns of the attackers. There are various algorithms for intrusion detection based on data mining. Most of the existing methods suffer from low accuracy and high false alarm rate. The proposed solution overcomes these drawbacks by improving the accuracy and reducing the false alarm rate. The system combines the clustering and classification techniques thereby forming a hybrid approach.
# TABLE OF CONTENTS

Abstract ........................................................................................................................................ ii

Table of Contents ........................................................................................................................ iii

List of Figures ................................................................................................................................... vi

List of Tables ................................................................................................................................... viii

1. Background and Rationale ........................................................................................................ 1
   1.1 Intrusion Detection System ................................................................................................. 1
       1.1.1 Misuse based Intrusion Detection System ............................................................... 1
       1.1.2 Anomaly based Intrusion Detection system ............................................................. 1
   1.2 Data Mining .......................................................................................................................... 2
   1.3 Data mining techniques for intrusion detection ................................................................. 2
       1.3.1 Classification ............................................................................................................. 2
       1.3.2 Clustering .................................................................................................................. 3
   1.4 Literature review on various Data mining techniques ...................................................... 4
   1.5 Existing Intrusion Detection systems .................................................................................. 5
       1.5.1 Intrusion detection system based on Boosted decision tree approach .............. 5
       1.5.2 Neural Network based Anomaly Intrusion Detection system ......................... 6
       1.5.3 Y-Means: A Clustering Method for Intrusion Detection ................................. 7
       1.5.4 Anomaly Detection using Support Vector Machine Classification with k-Medoids Clustering ................................................................. 9
       1.5.5 Intrusion Detection Algorithm Based On Semi-supervised Learning .......... 9

2. Narrative...................................................................................................................................... 11
   2.1 K-means Algorithm ............................................................................................................ 11
2.1.1 Algorithm ...........................................................................................................12
2.2 Problem Statement ............................................................................................12
2.3 Motivation ...........................................................................................................13
2.4 Project Objective ..............................................................................................13
3. Proposed System Design .......................................................................................14
  3.1 Modules of Proposed System ..........................................................................15
    3.1.1 Dataset ........................................................................................................15
    3.1.2 Clustering Module .......................................................................................15
    3.1.3 Outlier Detection .........................................................................................16
    3.1.4 Classification Module ................................................................................18
    3.1.5 Result Module .............................................................................................20
  3.2 Environment ......................................................................................................21
    3.2.1 Java Swing ................................................................................................21
    3.2.2 JFreeChart ..................................................................................................22
    3.2.3 Netbeans IDE .............................................................................................22
  3.3 System Configuration ........................................................................................23
    3.3.1 Hardware Configuration ............................................................................23
    3.3.2 Software Configuration .............................................................................24
4. Implementation and Results ..................................................................................25
  4.1 User Interface ....................................................................................................25
  4.2 K-Means Clustering ........................................................................................26
  4.3 Outlier Mechanism ...........................................................................................28
  4.4 Classification ....................................................................................................30

iv
LIST OF FIGURES

Figure 1. Stages of Intrusion Detection System.................................................................6
Figure 2. Y-Means Algorithm...............................................................................................7
Figure 3. Flow Diagram for Semi-Supervised Fuzzy Clustering Algorithm.....................10
Figure 4. Proposed System Design ....................................................................................14
Figure 5. User Interface ....................................................................................................25
Figure 6. Uploading KDD Dataset.......................................................................................26
Figure 7. Result for K-Means Clustered Data .................................................................28
Figure 8. Result for Outlier Detection ..............................................................................30
Figure 9. Result for Classification of Data .......................................................................31
Figure 10. Graphical Output .............................................................................................32
Figure 11. Comparative Result .........................................................................................33
Figure 12. Uploading Empty Text File .............................................................................37
Figure 13. Performing K-means on Empty.txt ...............................................................38
Figure 14. Screen Shot for Invalid Records ...................................................................38
Figure 15. Result Screen when file is not Uploaded.......................................................39
Figure 16. Performing K-means on Invalid Input ...........................................................40
Figure 17. Performing Outlier Mechanism on Invalid Input ..........................................40
Figure 18. Result for Invalid Input ..................................................................................41
Figure 19. Performing K-Means on Valid input ..............................................................42
Figure 20. Performing Outlier Mechanism on Valid input ............................................42
Figure 21. Performing Classification on Valid Input ..........................................................43
Figure 22. Result for Valid Input ..................................................43
Figure 23. Performing K-means on Corrected Data ..................................44
Figure 24. Performing Outlier Mechanism on Corrected Data ......................45
Figure 25. Result for Corrected Data ..................................................45
# LIST OF TABLES

Table 1. Sample Dataset ............................................................................................................27
Table 2. Sample Cluster 1 (DOS) ................................................................................................27
Table 3. Sample Cluster 2 (Probe) ..............................................................................................27
Table 4. Sample Cluster 3 (U2R) ..............................................................................................27
Table 5. Sample Cluster 4 (L2R) ..............................................................................................28
Table 6. Model construction ......................................................................................................29
Table 7. Outlier detection .........................................................................................................29
Table 8. Classification of data ...................................................................................................31
Table 9. Comparison with existing systems ..............................................................................34
1. BACKGROUND AND RATIONALE

1.1 Intrusion Detection System

With the development of network technologies, network crimes have also been increased rapidly which greatly affects the computer systems. Thus, network security is becoming an important aspect to protect the data from intruders and these attacks on networks are called as intrusions. To detect these malicious attacks, Intrusion detection system (IDS) are used [1]. Intrusion detection system is developed by using both hardware and software. Intrusion detection system monitors the network traffic automatically and detects the unusual patterns or intrusions that occur in the network. Based on the detection mechanism Intrusion detection systems are of two types, Misuse based IDS and Anomaly based IDS [1].

1.1.1 Misuse based Intrusion Detection System

Misuse based intrusion detection system detects the intrusions based on patterns. The system builds the patterns depending on various malicious activities and it detects the intrusions by looking at the known patterns [16]. The advantage of misuse based IDS is higher accuracy rate for known patterns and disadvantage is that intrusions can be detected only from the defined patterns [1].

1.1.2 Anomaly based Intrusion Detection system

Anomaly based Intrusion Detection system detects the intrusions based on the behavior of the networks and any deviation from this expected behavior are detected as attacks [16]. The advantage of anomaly based IDS is more complex and unknown intrusions can be detected and the disadvantage is high false alarm rate and low detection
rate [1]. This approach has high false alarm rate as every abnormal behavior is considered as attack but all deviations are not attacks.

1.2 Data Mining

Data mining is a process of knowledge extraction from large sets of databases [1]. Data mining can automatically detect patterns in a dataset that consists of large volumes of data and in the same data future instances can be detected by using these patterns. Intrusion detection system should monitor large volumes of data, where the data may be from different sources such as hosts, users, network etc which increases the data complexity. Analyzing complex data in huge network traffic is very difficult. Thus, the usage of data mining techniques is of great importance in intrusion detection system. With the help of data mining technology various patterns can be compared to identify their behavior and it can effectively separate normal data patterns from malicious data patterns.

1.3 Data mining techniques for intrusion detection

Data mining techniques play a vital role in intrusion detection system. Data mining techniques are used for the effective classification of abnormal patterns and normal patterns from large volumes of data over network. Different data mining techniques used to detect intrusions over the network are classification, clustering and association rules [1].

1.3.1 Classification

This is a data mining technique where each instance in a dataset is assigned to a
particular class. Important data classes are defined [15] to extract data models and these models are called as classifiers. In this technique learning and classification are two steps for data classification. In the learning step a classifier is formed and the class labels for the data are predicted by using this classifier. In the classification technique every data in the dataset has an attribute value that defines class and all the classes are predefined so that the analyst has a prior knowledge [1]. Classification can also be used to label every record in the data set and the records can be classified in predetermined set.

Classification is based on supervised learning i.e. it can only detect intrusions from label data. The major disadvantage of classification technique is that for the classification of datasets into normal and abnormal enormous amount of data need to be analyzed [1] as it cannot detect from unlabeled data. So, this technique is less efficient for intrusion detection and it is most commonly used for misuse detection.

1.3.2 Clustering

The data available over a network is very large, which makes human labeling time consuming and complex. Clustering a technique in which the data is assigned and labeled into groups of similar objects [1]. Every group is called as cluster and the members of same cluster are similar whereas different members from different clusters. Clustering is an unsupervised learning technique that can detect the intrusions from unlabeled data [15]. Thus, clustering techniques can be widely used for detecting complex intrusions. Clustering technique can be used for both misuse detection and anomaly detection.
1.4 Literature Review on various Data Mining Techniques

Kesavaraj G and Sukumaran S [6] presented a study on various classification techniques such as decision tree induction methods, rule-based methods, Bayesian network, neural network and support vector machines and his study states that one of the classification techniques cannot be chosen best among all and they depend on the dataset choice.

Jingke Xi [4] discussed different approaches for outlier detection in data mining perspective. Outlier detection algorithms are categorized into two types, classic outlier approach and spatial outlier approach. In classic outlier approach the outliers are analyzed depending on transaction dataset and in spatial outlier approach the outliers are analyzed based on spatial dataset.

Al Naqshbandi, S.M. and Samawi, V.W [3] have proposed a one rule genetic-fuzzy classifier system that uses Genetic algorithms to generate fuzzy rules, that is capable to detect the abnormal behavior over a network. The focus of the work is to increase the classification rate by evolving comprehensive rules, to produce short rules and with the increase in complexity of data automatic feature selection should be done.

Jiadong Ren et al. [5] have proposed an efficient outlier detection algorithm that partitions the heterogeneous data streams into chunks. After partitioning, the chunks are clustered and cluster references stores the result. To generate the final outlier references adjacent cluster references and representation degree are computed.

El-Semary et al. [2] proposed a system that uses fuzzy logic to detect intrusions. The system uses a fuzzy data mining algorithm to generate the rules when operating in
rule-generation mode and the rule subset is used as model for the input, finally the detection module uses the subset of rules to detect the intrusions.

Muda Z et al. [7] have proposed a hybrid approach i.e. KM+IR hybrid model by combining clustering and classification techniques to improve the false alarm rate. The approach combines K-means clustering algorithm with OneR classification and this approach have shown a decrease in false alarm rate.

Wang Huai-bin et al. [8] have proposed a hybrid algorithm called S-K hybrid model that uses K-means and self organizing maps (SOM) to overcome the drawback of SOM that fails to provide precise clustering results. At first the SOM neural network algorithm is used to train the process and to obtain the cluster centers and rough clusters, then K-means algorithm is used to refine the result produced from SOM stage.

Gupta et al. [16] have proposed a hybrid approach by integrating layered approach with conditional random fields (CRFs) to achieve high detection accuracy and high efficiency. This approach is robust and can handle noise data without affecting the performance.

### 1.5 Existing Intrusion Detection systems

#### 1.5.1 Intrusion detection system based on Boosted decision tree approach

This approach uses hoeffding tree classification technique to enhance the performance of intrusion detection system. In boosted decision tree the boosting is done by combining simple rules to form an ensemble [9] and thus the performance of a single ensemble is improved. The boosting algorithm at first trains all the data tuples by giving same weight and each tuple changes its weight according to the classification generated
by the classifier, and then the remaining classifiers are built by reweighting the tuple values. The final classification is formed by averaging all the classifications over the classifiers [9]. This method is a learning technique that combines various decision trees to form a classifier. Boosted decision tree approach has a good detection rate for intrusion detection.

Disadvantage

This approach combines various decision tree that make the system complex and it requires more space and time for processing and this approach has moderate false alarm rate.

1.5.2 Neural Network based Anomaly Intrusion Detection system

This technique uses a back propagation neural network to study the system behavior. Anomaly based intrusion detection system should detect the intrusions for dynamic change in behavior of data and by designing the intrusion detection system using artificial neural networks [10] the system can adapt to new environments. The system consists of four stages monitoring, detection, classification and altering.

![Figure 1. Stages of Intrusion Detection System](image)
The Figure 1 [10] shows the main stages of the intrusion detection system. The monitoring module is used as user interface for the system that provides tools to manage the processes in the system. The detection module detects the anomalies that follow five steps: feature selection, ranking, encoding, normalization, de-normalization and anomaly detection [10]. The classification module uses artificial neural networks for the classification of data and the final module alerting module identifies the abnormal behavior of the data and alerts the system administrator. This intrusion detection system detects the unknown anomalies and the system architecture can be easily modified and extended.

Disadvantages

This approach requires large volumes of data and it also require large amount of time to produce accurate results and this method requires increase in classification levels to increase detection percentage

1.5.3 Y-MEANS: A CLUSTERING METHOD FOR INTRUSION DETECTION

Y-means algorithm is a clustering algorithm similar to k-means algorithm used for intrusion detection. This technique partitions the data set automatically into reasonable clusters that classifies the instances into normal and abnormal clusters [11].

Figure 2 [11] describes the y-means algorithm. The y-means algorithm for intrusion detection first partitions the entire data into k-clusters, the clusters are formed by giving any integer between 1 and n, where n are total number of instances. After cluster formation the algorithm checks for empty clusters and if there are any empty clusters, these empty clusters are replaced with new clusters and the instances are
reassigned to the new existing clusters. This iteration continues until no empty cluster is discovered. This technique automatically generates the k values by splitting and merging the clusters and at last the clusters are labeled accordingly with their population. If the population ratio of a cluster is greater than threshold then it is labeled as normal otherwise all the instances in the cluster are labeled as intrusions.

![Y-means Algorithm Diagram](image)

**Figure 2. Y-means Algorithm**

*Disadvantage*

Though this technique has a better false alarm rate and detection rate it is not suitable for real time anomaly detection as the data sets cannot be updated dynamically during the process.
1.5.4 Anomaly Detection using Support Vector Machine Classification with k-Medoids Clustering

This technique is a hybrid learning approach that uses support vector machines for classification in replacement to naive bayes classifier. K-medoids is a partitioning algorithm for clusters that is similar to k-means [12]. This technique considers the centrally located data point in a cluster as called medoid or centroid in a cluster as reference point and the partitioning is done based on the distance metric. The algorithm initially determines the partitioning of the entire data set into number of clusters and it chooses a medoid arbitrarily for each cluster and the data object is clustered based on the medoid that is most similar with. After the cluster formation the support vector machine is trained and data classification is done to detect intrusions.

Disadvantages

This approach is not effective and the time complexity of the system is only reduced to an extent.

1.5.5 Intrusion Detection Algorithm Based On Semi-supervised Learning

This algorithm uses a semi-supervised fuzzy clustering technique to prevent the outlier sensitivity. The training data for the algorithm is a hybrid data of both labeled and unlabeled samples.

Figure 3 [13] illustrates the flow diagram for semi-supervised fuzzy clustering algorithm. The labeled data is used to initialize the cluster number (C) and to initialize the cluster centers and then the membership function is calculated. After the calculation of membership function the cluster base Ns are updated, then if the cluster base is greater
than threshold traverse the cluster and if the cluster base is less than threshold, split the cluster, update cluster number and calculate the new cluster formation center. The procedure is iterated until the cluster is stable and the output displays the type of data i.e. normal or abnormal.

Disadvantage

This approach is not suitable for large volumes of network data and it cannot detect various unknown intrusions.

Figure 3. Flow Diagram for Semi-Supervised Fuzzy Clustering Algorithm
2. NARRATIVE

The various clustering algorithms can be classified according to the creation of clusters of the objects. For our intention to use clustering algorithms in IDS, we need algorithms that can determine the jurisdiction of the object X to cluster, even if the object X was not included in the set of objects, from which we generate clusters. For this purpose in this project I chose the algorithms K-Means Clustering.

Algorithms of conceptual clustering are created by incremental way i.e. creating the structure of data by division of observed objects into subclasses. The result of these algorithms is a classification tree. Each node of the tree contains the objects of its child nodes, so root of this tree contains all objects. According to the above classification these algorithms are hierarchical, incremental algorithms that combine both – aggregation and division approach.

2.1 K-Means Algorithm

K-Means algorithm is a partitioned clustering algorithm. The main idea of the algorithm is to find K centers of clusters [14]. The question is, how effectively we can choose these centers of clusters, as the resulting clusters depend on the centers. The best approach would be, to pick center of cluster that is least similar to each other. The next step is to assign each object from data set to the center of cluster, to the most similar one. After this step, the next step in the classification is to determine the new center of each cluster (centers are derived from clusters of objects). Again the classification of objects into different clusters is performed according to their dissimilarity with new centers of
the clusters. These steps are repeated until we find the centers of clusters that no longer change or until it achieves maximum number of repetitions.

2.1.1 Algorithm

1. Define number of clusters ‘K’.

2. Initialize ‘k’ cluster centroid after verifying all centroids are different.

3. Iterate over all data points in the dataset, compute distance of centroids and assign nearest centroid to data point.

4. Re-calculate ‘k’ new centroids as per centers of clusters.

5. Repeat step-3 until centroids do not move or change distance.

2.2 Problem Statement

Most of the existing Intrusion detection systems have moderate accuracy and their true positive detection rate is low whereas false positive detection rate is high, this is because intrusion detection systems depend on network to detect intrusions and most of the intrusion detection systems follow single point of approach, insecure open network and exploitation of single nodes if strong attacks occur. Another problem is, the detection ratio and false alarm rate are affected when the attacks that are suspicious are considered as intrusions.
2.3 Motivation

Business organizations mostly rely on computer based information systems and these systems are connected to the network through internet, so the security of networks is considered as high importance to protect the data from intrusions. Thus, the need for an effective intrusion detection system is increasing with the increase in various types of network crimes that is simple with low false alarm rate.

2.4 Project Objective

The main objective of the proposed system is to reduce the false alarm rate by increasing the true positive rate and by decreasing false positive rate, to construct a simple system that has high accuracy. This is achieved by using both clustering and classification techniques that results in a hybrid model.
3. PROPOSED SYSTEM DESIGN

The proposed system consists of four modules: clustering module, outlier detection module, Classification module and result module. The KDD dataset is loaded and given as input to the clustering module where k-means clustering algorithm is applied to the dataset. The input from the clustering module is given to the outlier detection module where the outliers are detected, which is further classified by classification module and the result is displayed in the form of graph in result module. The overall system design of the proposed work is shown in Figure 4.

Figure 4. Proposed system design
The proposed system is a solution where the attributes defined for each data item is reduced as it follows a sequence of steps. This process ultimately results in improving the intrusion detection more efficient and also yields a less complex system with a better result.

3.1 Modules of Proposed System

3.1.1 Dataset

In our proposed system, we use KDDcup99 Dataset [18] as input. The dataset is static, that is provided for intrusion detection. The KDD dataset is composed with various types of attacks that has different testing and training data. The dataset contains 24 training attack types and has additional 14 types in test data [23]. The dataset has simulated attacks that fall in one of the following four attack categories: Denial of Service Attack (DOS). Probe attack, User to Root Attack (U2R), Remote to Local attack(R2L). The example of a sample attack in a dataset is shown below:

13,tcp,telnet,SF,118,2425,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00,0.00,0.38,0.12,0.04,0.00,0.00,0.00,0.12,0.30,guess_passwd,2 [18].

3.1.2 Clustering Module (K-Means)

In this module K-Means clustering algorithm is applied for the dataset to classify it into k clusters. K-Means algorithm is one of the easiest and simplest ways of partitioning large sets of data into clusters. The KDD dataset is given as input to the K-Means clustering module.
The dataset is processed into clusters by using the K-Means algorithm and it is an iterative process that classifies k different clusters by converging to a local minimum. The clusters formed based on their similarities, and then the centroids are computed for each data item. All the nearest items are grouped or clustered until all the attributes are completed. If the attribute changes its position during the process, then all those changed attributes are re-clustered and the clusters are updated again. The change in attribute position occurs when the dataset is changed intermediately and the proposed system is a continuous process so the change in dataset is flagged before clustering.

3.1.3 Outlier Detection

Spatial domain consists of different objects and each object has different position with the other objects in the system. The outlier detection module performs the algorithm on the objects with their position values to find the maximum deviation of objects with its neighbors within the system. The algorithm is performed in two steps, the first step is model construction in which the models are identified for each attribute and each model is mapped with the ID of each attribute for all the clusters. The second step is outlier detection, in which the outliers are detected i.e. the objects with more deviation in values are spotted.

In the model construction step, the clustered data is taken as input given and signal attribute value is calculated based on the properties of config.info. If the signal attribute value is equal to the property value then attack type is categorized. After identifying all the attacks the attack attribute and attack type are added to a vector.

In the outlier detection step, array lists are declared for each type of attack. The default threshold is read form the configure.info and default strength is assumed as 0.
Each signal attributes are taken from the list of signal attributes. The strength of a signal is updated when the calculated signal attribute value is equal to the property value. By taking the max and min values from the clusters the maximum distance is calculated and the signal with larger distance is detected as outlier.

Model construction is a process of identifying the signals. The algorithm [16] steps outlier detection are shown below.

**Model Construction:**

1. Take ‘k’ clusters as input.
2. Read each cluster.
3. Identify model in the cluster for each attribute (here model is a signal attack).
4. Map each model with the ID of each attribute
5. Redo 3rd and 4th step until all attributes are completed
6. Redo 2nd step until all clusters completed

**Outlier Detection:**

1. Take attributes already model constructed.
2. Read model flagged clusters.
3. Calculate the distance based on min and max values
4. Update the strength of each model.
5. Read configuration to pass the attributes to find the Threshold.
6. Redo steps until all models completed
7. Generate output as processed data.
Example:

For example let the list consists of 8 signals 0, 1, 2, ….8, in array list and each signal attribute is taken from the list of signals, and property attribute list consists of signal key and its value, these values are retrieved and stored. The attribute value is calculated for signal 0 and here the value is 1. If the attribute value is equals the configuration properties value then the status is will be yes and also strength will increased by 1. In the outlier detection the distance is calculated by considering min and max values using the formula \( \text{dist} = \sqrt{((t1-t3)*(t1-t3)) + ((t2-t4)*(t2-t4))} \). The strength is updated and the threshold value is obtained from configure.info. Depending on all the values the signal is considered as an outlier and categorized. The results show the type of attack.

3.1.4 Classification Module

The classification module uses the concept of fuzzy rules and fuzzy classification is done based on these rules are generated by grouping the attributes into different classes. The maximum and minimum deviation values generate the definite and indefinite rules, which are further used to calculate the information gain of each attribute. The fuzzy rules are applied to simply the classification process; the rules are applied after outlier mechanism. The basic fuzzy rules are generated depending on the nature of the signal, i.e. duration of signal, service, number of time the signal occurrence etc. The fuzzy rule application is shown as follows.
1. Fuzzy rules

Take outlier model of each attribute.

While (Outlier_Model.Model_ID)

If(outlier_model=Fuzzy_PT) then

If(Outlier_Value=Fuzzy.Deviation_Type) then

Model_ID.Deviation:= Config.Deviation

Else

Model_ID.Deviation:=Config.Normal

End_if

End_if

End_while

To classify the signals based on all the signal attributes is a very difficult task, to simplify the classification process fuzzy rules and configuration properties are defined with fewer attributes. Thus system complexity is reduced due to this reduction in attributes [18]. The steps for fuzzy classification are explained below:

- Read deviated and normal models \[ C_{2d\_range} = [1, 1e2, 1e4] \] [26]

  We are taking complete input list to perform the operations

- While (Model_ID.Attribute_ID) \[ \text{for C in } C_{2d\_range} \] [26]

  We are performing a while loop to perform operations on models constructed and we already have four categories of attacks to perform the operations so this looping \( \text{(for gamma in gamma}_{2d\_range}\text{)} \) is not considered.

- If(Model_ID.Deviation>= Fuzzy.Scorer) then
Attribute_ID.Attribute_type:=Config.Attribute_val
Attribute_ID.Attack_type:=Config.Attack_Type

The deviation values are computed for each signal and these deviation values are checked with fuzzy scorer and a decision is taken and the process continues until all signals are classified.

2. Fuzzy classification Pseudocode

Read deviated and normal models

While (Model_ID.Attribute_ID) do

    If(Model_ID.Deviation>= Fuzzy.Scorer) then

        Attribute_ID.Attribute_type:=Config.Attribute_val
        Attribute_ID.Attack_type:=Config.Attack_Type

    Else

        Attribute_ID.Attribute_type:=Config.NORMAL

    End_if

End_while

3.1.5 Results Module

The results module displays the result in the graphical format for the analyzed dataset. The graphical format displays the True positive (TP), True Negative (TN), false positive (FP) and False negative (FN) of the data. The graphical representation is a bar
chart that shows the calculated accuracy, sensitivity and specificity of the system and the results are compared with existing CRF model [17]. The calculation of accuracy, sensitivity and specificity of the system are performed using the following equations.

Accuracy of the system: \( \frac{TN+TP}{TN+TP+FN+FP} \)

Sensitivity of the system: \( \frac{TP}{TP+FN} \)

Specificity of the system: \( \frac{TN}{TN+FP} \)

3.2 Environment

The proposed system is implemented in Java and the environment used is Netbeans IDE. Java swing and JFreeChart are mainly used for the development.

3.2.1 Java Swing

Swing is a Java Graphical interface widget tool kit which is an API for building user interfaces for Java programming [21]. Java swing is developed by oracle and it is part of Java foundation classes (JFC). It is accomplished by more sophisticated GUI components and it is developed to replace Abstract Windowing Toolkit (AWT). Abstract Windowing Toolkit is platform dependent i.e. the programs written in AWT have different behavior in different platforms, but swing is completely developed using java, hence the programs written in swing behaves same in any platform. Swing components are not only platform dependent but they are customizable, extensible, configurable and lightweight [21]. Swing has similar components such as labels, buttons and checkboxes but it also includes advanced components such as tables, lists, panels and scroll panes. The swing class names start with J such as JPanel, JButton and JList. Some of the top-level classes of swing are heavy-weighted as they’re derived from AWT.
3.2.2 JFreeChart

JFreeChart is an open source Java chart library that helps to develop various number of professional charts [24]. The extensive features of JFreeChart include a well-documented and persistent API that support a large set of chart types, followed by a flexible design, which can be targeted to both the server-side and the client-side applications, support for many types of outputs like image files (includes PNG and JPEG), vector graphics file formats (includes EPS, PDF and SVG) and Swing components. JFreeChart is an open source that is distributed under the terms of GNU Lesser general Public License (LGPL), which is further used in proprietary application.

In JFreeChart map visualizations can be done by different values that relate to the geographical areas. Examples of this include density of population in each state in US, capital for various incomes for each country in Europe, and life expectancy of each country in the world. Time Series Chart Interactivity is another new feature to JFreeChart, which is used to display time series data. Another feature includes Dashboards, in which a mechanism is created that is flexible and supports a subset of JFreeChart and can be delivered easily using both applet and Java Web Start. At last, property editors are the mechanisms included in JFreeChart that can handle the subsets of properties to set charts. Extension of this mechanism can be done for providing efficient end-user control on the appearance of the charts.

3.2.3 Netbeans IDE

Netbeans is an integrated development environment that offers various tools for
enterprise, desktop, mobile application and java web development [22]. Netbeans is an open source framework that not only supports Java but various languages such as C, C++, PHP and HTML. It is developed in Java and can run on multiple operating systems such as windows, Linux, Solaris and other platforms that are compatible with Java virtual machine (JVM). The project is implemented in Netbeans IDE 8.0. Netbeans IDE 8.0 is the latest version that provides out-of-the box code analyzers and editors for Java 8 technologies [21]. The Netbeans IDE has various features such as Wizard framework that supports step-by-step dialogs, Netbeans Visual Library, Integrated development tools, User interface management, User settings management, Storage management and Window management. Netbeans 8.0 is enhanced to provide a better support for Maven and Java EE and the support tools are also enhanced to deploy, run and debug.

3.3 System Configuration

3.3.1 Hardware Configuration

- Processor - Intel core i3
- Speed - 1.1 Ghz
- RAM - 4 GB(min)
- Hard Disk - 20 GB
- Key Board - Standard Windows Keyboard
- Mouse - Two or Three Button Mouse
3.3.2 Software Configuration

- Operating System : Windows XP
- Programming Language : JAVA.
- Java Version : JDK 1.7 & above.
- IDE : NetBeans
- Dataset : KDD Dataset
4. IMPLEMENTATION AND RESULTS

4.1 User Interface

User interface consists of five functionalities namely K-Means cluster, Outlier Mechanism, Classification, Result and Load. Figure 5 shows the user interface.

![User interface screenshot](image)

Figure 5. User interface

The Load functionality is used to load the dataset and after the dataset is loaded other functionalities can be applied. Figure 6 shows the loading of KDD dataset.
4.2 K-Means clustering

K-Means clustering algorithm is applied for the KDD data set by clicking this button and clusters are formed. The table 1 shows the sample KDD dataset set with 41 features and different classes of intrusion categories, with the application of K-Means algorithm clusters are formed that covers four major categories of attacks i.e. DOS, Probe, U2R and R2L. Table 2, Table 3, Table 4 and Table 5 illustrates the cluster formation for major attacks that are sub-categorized into other attacks.

Figure 6. Uploading KDD Dataset
Table 1. Sample Dataset

<table>
<thead>
<tr>
<th>ID</th>
<th>Name of Attribute</th>
<th>Types of Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>guess_passwd</td>
<td>NA</td>
</tr>
<tr>
<td>2</td>
<td>Snmpguess</td>
<td>NA</td>
</tr>
<tr>
<td>3</td>
<td>Processtable</td>
<td>NA</td>
</tr>
<tr>
<td>4</td>
<td>Normal</td>
<td>Normal</td>
</tr>
<tr>
<td></td>
<td>:</td>
<td>:</td>
</tr>
<tr>
<td></td>
<td>23700</td>
<td>Snmpguess</td>
</tr>
</tbody>
</table>

Table 2. Sample Cluster 1 (DOS)

<table>
<thead>
<tr>
<th>ID</th>
<th>Name of Attribute</th>
<th>Types of Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Back</td>
<td>DOS</td>
</tr>
<tr>
<td>2</td>
<td>Neptune</td>
<td>DOS</td>
</tr>
<tr>
<td>3</td>
<td>Neptune</td>
<td>DOS</td>
</tr>
<tr>
<td>4</td>
<td>Back</td>
<td>DOS</td>
</tr>
</tbody>
</table>

Table 3. Sample Cluster 2 (Probe)

<table>
<thead>
<tr>
<th>ID</th>
<th>Name of Attribute</th>
<th>Types of Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Nmap</td>
<td>Probe</td>
</tr>
<tr>
<td>2</td>
<td>Satan</td>
<td>Probe</td>
</tr>
<tr>
<td>3</td>
<td>Satan</td>
<td>Probe</td>
</tr>
<tr>
<td>4</td>
<td>Satan</td>
<td>Probe</td>
</tr>
</tbody>
</table>

Table 4. Sample Cluster 3 (U2R)

<table>
<thead>
<tr>
<th>ID</th>
<th>Name of Attribute</th>
<th>Types of Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>buffer_overflow</td>
<td>U2R</td>
</tr>
<tr>
<td>2</td>
<td>Rootkit</td>
<td>U2R</td>
</tr>
<tr>
<td>3</td>
<td>buffer_overflow</td>
<td>U2R</td>
</tr>
<tr>
<td>4</td>
<td>buffer_overflow</td>
<td>U2R</td>
</tr>
</tbody>
</table>
### Table 5. Sample Cluster 4 (R2L)

<table>
<thead>
<tr>
<th>ID</th>
<th>Name of Attribute</th>
<th>Types of Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Warezmaster</td>
<td>R2L</td>
</tr>
<tr>
<td>2</td>
<td>Warezmaster</td>
<td>R2L</td>
</tr>
<tr>
<td>3</td>
<td>Warezmaster</td>
<td>R2L</td>
</tr>
<tr>
<td>4</td>
<td>Warezmaster</td>
<td>R2L</td>
</tr>
</tbody>
</table>

Figure 7 shows the screenshot of clustered data after performing K-Means algorithm.

Figure 7. Result for K-Means Clustered Data

#### 4.3 Outlier Mechanism

The outlier mechanism is divided into two steps. In the first step model is constructed and in the second step outlier is detected. The input to the outlier detection is the K-Means clustered data. Table 6 shows the model construction for clustered data.
where each cluster is read, model for each attribute is identified and model is mapped with the ID of each attribute.

Table 6. Model construction

<table>
<thead>
<tr>
<th>ID</th>
<th>Model</th>
<th>Model Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>27</td>
<td>Back</td>
<td>NA</td>
</tr>
<tr>
<td>263</td>
<td>Neptune</td>
<td>NA</td>
</tr>
<tr>
<td>1345</td>
<td>Ps</td>
<td>NA</td>
</tr>
<tr>
<td>2981</td>
<td>Satan</td>
<td>NA</td>
</tr>
<tr>
<td>476</td>
<td>buffer_overflow</td>
<td>NA</td>
</tr>
<tr>
<td>1915</td>
<td>Rootkit</td>
<td>NA</td>
</tr>
<tr>
<td>28</td>
<td>warezmaster</td>
<td>NA</td>
</tr>
<tr>
<td>430</td>
<td>Multihop</td>
<td>NA</td>
</tr>
</tbody>
</table>

After model construction, outliers are to be detected. The model is read and strength is calculated for each model. The outliers (Threshold) are identified by passing the configuration to attributes and the data is processed. Table 7 shows the data for outlier detection, where the ID's 27,28......2891 are the outliers. The outliers are calculated based on the average values between neighbors and the strengths are updated based on the attributes in the attack. The strength represents the deviation of the attack or signal from the normal position. The Threshold values in the column represent the occurrence of the attack.

Table 7. Outlier detection

<table>
<thead>
<tr>
<th>ID</th>
<th>Strength</th>
<th>Threshold</th>
</tr>
</thead>
<tbody>
<tr>
<td>27</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>28</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>263</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>430</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>476</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>
Figure 8 shows the screenshot of results after model construction and outlier detection.

<table>
<thead>
<tr>
<th>ID</th>
<th>Model</th>
<th>Model Type</th>
<th>Strength</th>
</tr>
</thead>
<tbody>
<tr>
<td>1345</td>
<td></td>
<td>N/A</td>
<td>1</td>
</tr>
<tr>
<td>1915</td>
<td></td>
<td>N/A</td>
<td>1</td>
</tr>
<tr>
<td>2891</td>
<td></td>
<td>N/A</td>
<td>1</td>
</tr>
</tbody>
</table>

4.4 Classification

In the classification model fuzzy rules are applied to the outlier model of each attribute and by using fuzzy classification the data is classified further. Table 8 shows the data after the application of fuzzy classification, the results represent the ID, the attack occurred, and the score of the attack. The score of the attack represents the strength of the attack.
Table 8. Classification of data

<table>
<thead>
<tr>
<th>ID</th>
<th>Attack Type</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>27</td>
<td>Back</td>
<td>2</td>
</tr>
<tr>
<td>263</td>
<td>Neptune</td>
<td>1</td>
</tr>
<tr>
<td>1345</td>
<td>Ps</td>
<td>1</td>
</tr>
<tr>
<td>2981</td>
<td>Satan</td>
<td>1</td>
</tr>
<tr>
<td>476</td>
<td>buffer_overflow</td>
<td>1</td>
</tr>
<tr>
<td>1915</td>
<td>Rootkit</td>
<td>1</td>
</tr>
<tr>
<td>28</td>
<td>Warezmaster</td>
<td>1</td>
</tr>
<tr>
<td>430</td>
<td>Multihop</td>
<td>1</td>
</tr>
</tbody>
</table>

Figure 9 shows the screenshot after the application of fuzzy classification.

Figure 9. Result for Classification of Data
4.5 Result

The result module displays the True positive, False positive, True negative and False negative values in a graphical format and comparison results with the existing CRF model is displayed. Figure 10 shows the graphical result and Figure 11 shows the comparison of proposed system with the CRF model.

Figure 10. Graphical Output
4.6 Results and Discussion

The proposed system uses KDD dataset for training and testing. The KDD dataset consists of 24 types of attacks for training and it has more 14 attack types in test data. The system is combining K-means algorithm with classification technique to form a hybrid method for detecting intrusions. The outlier detection mechanism simplifies the data further for classification and with the application of fuzzy classification helps to achieve maximum detection accuracy. The performance of the system is obtained by calculating accuracy, sensitivity and specificity. The system is trained and tested with 94,800 instances. The system when tested have better performance in detecting the DOS, Probe, U2R and R2L attacks with an accuracy of 100%. The system sensitivity i.e. detection rate is 100% and the specificity of the system is 87%. The proposed system is compared with other hybrid methods that use the KDD dataset for testing, they are
layered approach with conditional random field (CRF) model and S-K hybrid model. Table 9 shows the comparison between the models.

Table 9. Comparison with existing systems

<table>
<thead>
<tr>
<th>Methods</th>
<th>Proposed system</th>
<th>Existing CRF model</th>
<th>S-K hybrid model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>100%</td>
<td>66%</td>
<td>-</td>
</tr>
<tr>
<td>Sensitivity(DR)</td>
<td>100%</td>
<td>88%</td>
<td>92%</td>
</tr>
<tr>
<td>Specificity</td>
<td>87%</td>
<td>73%</td>
<td>65%</td>
</tr>
</tbody>
</table>

The proposed system is achieving high accuracy as it is tested with training data and the comparison results show that the proposed system has a higher accuracy and detection rate than other models. The system has enhanced the detection rate i.e. it has an increase of +12% to the CRF model and an increase of +8% to the S-K model. The system has an increase in accuracy rate up to +34% to the CRF model. The specificity of the system is also improved when compared with other models.
5. TESTING AND EVALUATION

5.1 System Testing

The main need for testing is to detect errors. It is the process of discovering errors, which are mostly probable weakness or fault of the proposed system. The functionality of various components, assemblies and final products can be checked through testing. With the process of testing, we can make sure that the software of the system meets the requirements based on the user expectations. It also checks for the system’s case of failing in any unacceptable way. We have different types of testing, where each kind represents a specific testing requirement.

5.2 Types of Testing

5.2.1 Unit testing

The designing of unit testing involves the process of validating the internal programming logic’s functionality, along with the program inputs and the corresponding outputs. The validation also involves the internal code flow and all the decision branches. Unit testing involves the testing of individual software applications, which is done before integration. It is a structural testing, which relies on the knowledge of construction and its invasion. Unit testing ensures a unique path for each business process to perform specifically for the documented specifications, which defines clear inputs and the expected results.

5.2.2 Integration testing

Integration testing is performed for testing integrated software components to
check if they are running as one program. The testing process done in unit testing shows the successful results with individual components, whereas in integration testing, the correctness and consistency of the combination of components is checked. Thus, this type of testing is mainly done to check the problems that raise with the combination of different components.

5.2.3 Functional testing

In functional testing, the functionality of specified functions demonstrated based on some technical and business requirements, user manuals and system documentation. Functional testing mainly concentrates on valid and invalid inputs, functions of the system followed by the output. Besides these, functional testing also focuses on key functions, various requirements, few special test cases, predefined and successive processes, data fields and business process flows. Before performing the functional testing, additional test are identified and the corresponding effective values of the current test are considered.

5.2.4 System Testing

System testing is performed so as to check if the system meets all the mentioned requirements. It also tests for ensuring the configuration of known and predictable results. Configuration oriented system integration test is considered as an example for system testing. This type of testing is based on integration points, process description and flows and on emphasizing pre-driven process links. System testing provides various inputs along with the outputs without considering the working structure of the software.
5.3 Test cases

The project is tested at system level by giving various inputs to the system such as valid inputs, invalid inputs and performance of the system is observed. The following are the test cases for the proposed system.

**Test case 1: Uploading Empty text file**

When the system is given an empty text file as input, the K-Means cluster performs the K-means algorithm and no clusters are formed. If the user further clicks to perform Outlier mechanism a message is displayed on the screen that no valid records are found. Figure 12 shows the screenshot for uploading an empty file.

![Figure 12. Uploading Empty Text File](image)

Figure 13 shows the screenshot of performing K-Means on empty file
Figure 13. Performing K-means on empty.txt

Figure 14 shows the screen shot for invalid records.

Figure 14. Screen Shot for Invalid Records
**Test Case 2: Uploading No File**

The system should not work if any input file is not loaded. If the system is not provided with any input file and if the user is trying to apply the algorithms the system should display an invalid message which shows the proper working of system functionality. Figure 15 shows the screen short for no input data.

![Figure 15. Result Screen when file is not Uploaded](image)

**Test Case 3: Uploading invalid input**

If the system is given invalid input i.e. the file that has no malicious data then the system should provide no results as any intrusions are present and it should display a message that there are no records to perform. Figure 16 shows the screenshot of performing K-means on Invalid dataset.
Figure 16. Performing K-means on Invalid Dataset

Figure 17 shows the screenshot of performing Outlier mechanism on Invalid dataset.

Figure 17. Performing Outlier Mechanism on Invalid Dataset

Figure 18 shows the result for Invalid input.
Test Case 4: Uploading valid input

If the system is given a valid input, it should perform all the algorithms and the detected intrusions should be displayed in the graphical format. Figure 19 shows the screenshot for performing K-means on valid input, Figure 20 shows the screenshot for performing outlier detection on valid input, Figure 21 the screenshot for performing clustering on valid input and Figure 22 shows the result for valid input.
Figure 19. Performing K-Means on Valid Input

Figure 20. Performing Outlier Mechanism on Valid Input
Figure 21. Performing Clustering on Valid Input

Figure 22. Result for Valid Input
Test case 5: uploading corrected dataset

The system is uploaded with corrected dataset and the results are shown below. Figure 23 shows the screenshot of performing K-means on corrected data.

Figure 23. Performing K-means on Corrected Data

Figure 24 shows the screenshot of performing outlier mechanism on corrected data and Figure 25 shows the screenshot of result for corrected data.
Figure 24. Performing Outlier Mechanism on Corrected Data

Figure 25. Result for corrected data
6. CONCLUSION AND FUTURE WORK

A hybrid model was developed by combing clustering and classification techniques for intrusion detection. The system combines K-means algorithm with radial support vector machines to improve the accuracy of the system and to reduce the false alarm rate. The K-Means clustering algorithm used in the system handles the heterogeneity of the data more efficiently. Outlier detection algorithm was adopted to detect outliers which simplify the data for classification. Fuzzy rules and fuzzy classification is used to classify the data and results are displayed in a graphical format. This system performs intrusion detection in a simpler way by reducing the time complexity of the system and the system when compared with a conventional model achieved a good accuracy and the false alarm rate is also decreased. Thus the intrusion detection can be done more efficiently using proposed system.

The future work can be done to develop a system that detects the unlabeled data effectively with less time complexity and further work can be done to train and test the system with different datasets.
BIBLIOGRAPHY AND REFERENCES


47


[22] https://netbeans.org/features/


